MULTIPROGRAMMING OPERATING SYSTEM (MOS) PROJECT

Second Phase

ASSUMPTIONS:

* Jobs may have program errors
* PI interrupt for program errors introduced
* No physical separation between jobs
* Job outputs separated in output file by 2 blank lines
* Paging introduced, page table stored in real memory
* Program pages allocated one of 30 memory block using random number generator
* Load and run one program at a time
* Time limit, line limit, out-of-data errors introduced
* TI interrupt for time-out error introduced
* 2-line messages printed at termination

NOTATION

M: memory

IR: Instruction Register (4 bytes)

IR [1, 2]: Bytes 1, 2 of IR/Operation Code

IR [3, 4]: Bytes 3, 4 of IR/Operand Address

M[&]: Content of memory location &

IC: Instruction Counter Register (2 bytes)

R: General Purpose Register (4 bytes)

C: Toggle (1 byte)

PTR: Page Table Register (4 bytes)

PCB: Process Control Block (data structure)

VA: Virtual Address

RA: Real Address

TTC: Total Time Counter

LLC: Line Limit Counter

TTL: Total Time Limit

TLL: Total Line Limit

EM: Error Message

← : Loaded/stored/placed into

INTERRUPT VALUES

SI = 1 on GD

= 2 on PD

= 3 on H

TI = 2 on Time Limit Exceeded

PI = 1 Operation Error

= 2 Operand Error

= 3 Page Fault

Error Message Coding

EM Error

0 No Error

1 Out of Data

2 Line Limit Exceeded

3 Time Limit Exceeded

4 Operation Code Error

5 Operand Error

6 Invalid Page Fault

BEGIN

INITIALIZATION

SI = 3, TI = 0

MOS (MASTER MODE)

Case TI and SI of

TI SI Action

0 1 READ

0 2 WRITE

0 3 TERMINATE (0)

2 1 TERMINATE (3)

2 2 WRITE, THEN TERMINATE (3)

2 3 TERMINATE (0)

Case TI and PI of

TI PI Action

0 1 TERMINATE (4)

0 2 TERMINATE (5)

0 3 If Page Fault Valid, ALLOCATE, update page Table, Adjust IC if necessary,

EXECUTE USER PROGRAM OTHERWISE TERMINATE (6)

2 1 TERMINATE (3,4)

2 2 TERMINATE (3,5)

2 3 TERMINATE (3)

READ

If next data card is $END, TERMINATE (1)

Read next (data) card from input file in memory locations RA through RA + 9

EXECUTEUSERPROGRAM

WRITE

LLC ← LLC + 1

If LLC > TLL, TERMINATE (2)

Write one block of memory from locations RA through RA + 9 to output file

EXECUTEUSERPROGRAM

TERMINATE (EM)

Write 2 blank lines in output file

Write 2 lines of appropriate Terminating Message as indicated by EM

LOAD

LOAD

While not e-o-f

Read next (program or control) card from input file in a buffer

Control card: $AMJ, create and initialize PCB

ALLOCATE (Get Frame for Page Table)

Initialize Page Table and PTR

Endwhile

$DTA, STARTEXECUTION

$END, end-while

Program Card: ALLOCATE (Get Frame for Program Page)

Update Page Table

Load Program Page in Allocated Frame

End-While

End-While

STOP

STARTEXECUTION

IC ← 00

EXECUTEUSERPROGRAM

END (MOS)

EXECUTEUSERPROGRAM (SLAVE MODE)

ADDRESS MAP (VA, RA)

Accepts VA, either computes & returns RA or sets PI ← 2 (Operand Error) or PI ← 3 (Page Fault)

LOOP

ADDRESSMAP (IC, RA)

If PI ≠ 0, End-LOOP (F)

IR ← M[RA]

IC ← IC+1

ADDRESSMAP (IR[3,4], RA)

If PI ≠ 0, End-LOOP (E)

Examine IR[1,2]

LR: R ← M [RA]

SR: R → M [RA]

CR: Compare R and M [RA]

If equal C ← T else C ← F

BT: If C = T then IC ← IR [3,4]

GD: SI = 1 (Input Request)

PD: SI = 2 (Output Request)

H: SI = 3 (Terminate Request)

Otherwise PI ← 1 (Operation Error)

End-Examine

End-LOOP (X) X = F (Fetch) or E (Execute)

SIMULATION

Increment TTC

If TTC = TTL then TI ← 2

If SI or PI or TI ≠ 0 then Master Mode, Else Slave Mode